ESSENCE POCKET GUIDE

A quick introduction to Essence

IVAR JACOBSON INTERNATIONAL

Creating winning teams.

Essence Pocket Guide

Introduction

Essence is a standard for the creation, use and improvement of software engineering practices and methods, which is maintained and published by the OMG international open standards consortium.

The spirit of Essence is to concentrate on the essential information and to optimize both the technical and human aspects of engineering by providing super-lightweight practices, often distilled into a small handful of cards, that focus on outcomes and minimize production of documentation.

As an industry standard, Essence describes a **language** and a **kernel** for these engineering practices:

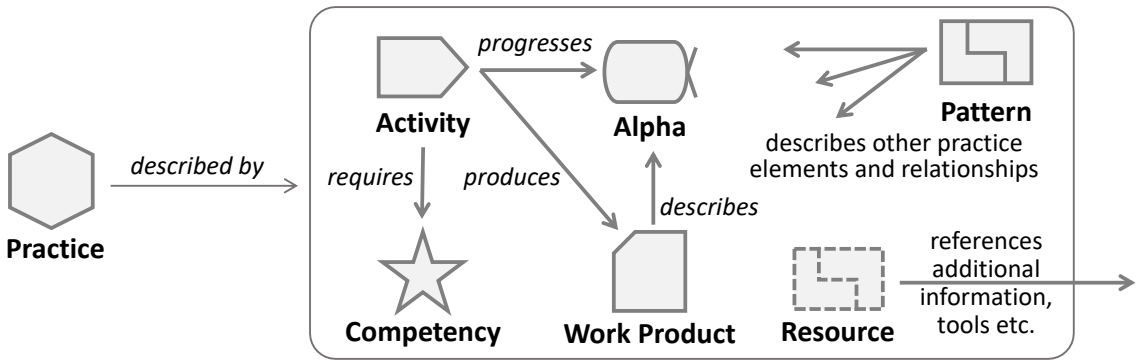
- **The Essence Language** enables practices to be expressed in a simple and standard form that ensures that they can be easily shared, understood, and applied both independently and in combination with other Essence practices.
- **The Essence Kernel** provides the common ground for defining these Essence practices. It includes the essential elements that are always central to every software engineering endeavor. The Kernel also helps practitioners compare practices and make informed decisions about which practices to adopt, and how to apply and adapt them.

The Essence Language

A **Practice** is a repeatable approach to doing something with a specific purpose in mind. It can be used independently or in combination with other practices. Examples might include *User Stories*, *Use Cases* or *Scrum*.

A practice "Tells a story" of how we achieve some valuable outcomes. It typically has:

- One or more Activities, that progress …
- one or more key elements called Alphas, that are captured and communicated using …
- one of more Work Products, the whole being supplemented and joined-up with …
- one or more Patterns, and more information is sign-posted, or sources cited, using …
- references to one or more Resources.



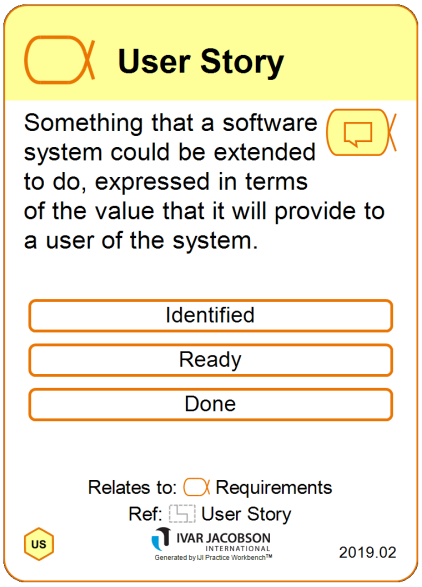
A practice is described by a set of these different kinds of items, which will be described briefly in the rest of this guide, along with examples of each one.

Alphas

An **Alpha is** a key aspect or element that we need to progress, the state of which is a key indicator of the overall progress and health of an endeavor. Examples might include a *User Story* being progressed to a *Done* state *or a Team* achieving a state of *Performing*.

Alphas are what we progress. These are of central importance in Essence because they ensure we remain focused on the valuable outcomes we are trying to achieve, not on secondary concerns such as what physical documents or other artefacts may or may not help us to achieve these outcomes.

It progresses through a number of states, from top to bottom, as shown in this example of a User Story.



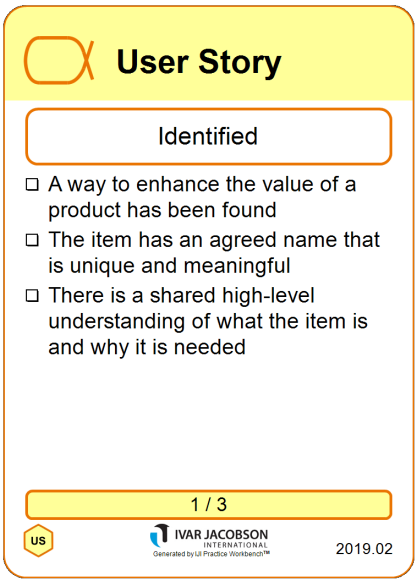
Alpha States

A **State** is a specification of the state of progress of an Alpha. Examples might include a *User Story* Alpha being in a State of *Identified* or an *Impediment* being *Resolved*.

Alpha States can in turn be concisely and accurately defined in terms of the set **Checklist Items** that need to be achieved for the state to be achieved.

In this example, the *Identified* State of a *User Story* Alpha might have these 3 Checklist Items that should be satisfied before it can be considered in that State.

These Checklist Items can be used both as a way to assess whether an item is in this state, or considered as a set of "To Dos" to achieve it.

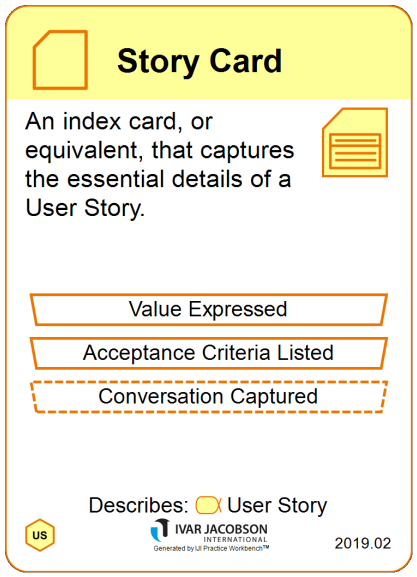


Work Products

A **Work Product** is an artifact of value and relevance for a software engineering endeavor. Examples of these kinds of physical artefacts might include a *Story Card* or a *Kanban Board*.

A Work Product is primarily defined in terms of the Levels of Detail that are captured by it, from top to bottom.

Some Levels of Detail may be optional (indicated by a dashed outline) as shown here for the example level *Conversation Captured.* It is not required for all *Story Cards* but may be captured if it adds value.



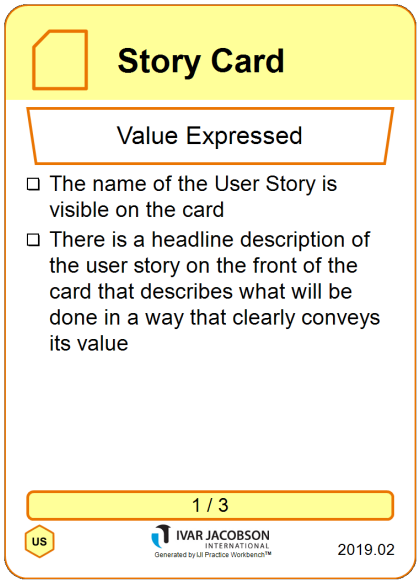
Work Product Levels of Detail

A **Level of Detail** is the amount of detail or range of content in a Work Product. Examples might include a *Story Card* having its *Value Expressed* or additionally having its *Acceptance Criteria Listed*.

Just as Alphas have States which describe how far they have been progressed, so Work Products have Levels of Detail which describe how much and what kind of information has been captured.

Work Product Levels of Detail can also be concisely and accurately defined in terms of a set **Checklist Items**. For example, the *Value Expressed* Level of Detail for a *Story Card* has these 2 items which should be achieved.

Once again, these Checklist Items can be used both as a way to assess an item or considered as a set of "To Dos" to achieve this Level of Detail.

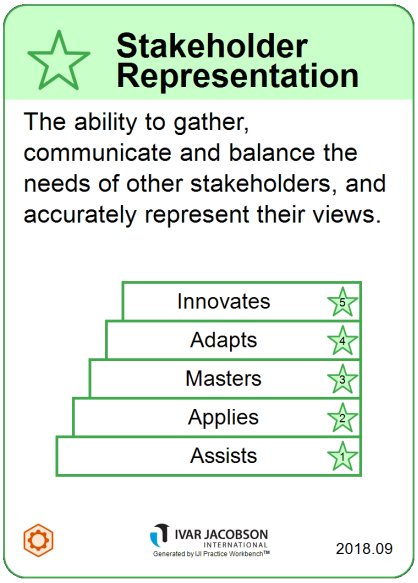


Competencies

A **Competency** encompasses the abilities, capabilities, attainments, knowledge and skills necessary to do a certain kind of work, such as *Leadership*, *Development* or *Testing*.

Teams are central and critical to the success of endeavors and Activities. In particular, it is important to understand what kind of team members we need, with what kinds of Competencies.

In this *Stakeholder Representation* example the core capabilities are described, with a number of Competency Levels possible.



Competency Level

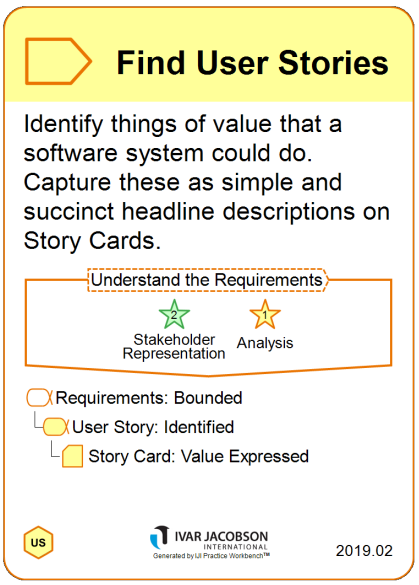
A **Competency Level** defines what level of capability a team member has with respect to a Competency, from level 1 *Assists* to Level 5 *Innovates*.

Activities

An **Activity** is the doing of some work by one or more people, possibly in a workshop or meeting. Examples might include *Find User Stories, Daily Stand-Up*, or *Backlog Refinement*.

A good Activity description will include guidance on:

- What outcomes the Activity produces or achieves in terms of Alpha States or Work Product Levels of Detail
- What we should do to achieve these outcomes
- What kinds of Competencies at what Levels are needed to perform this Activity successfully.



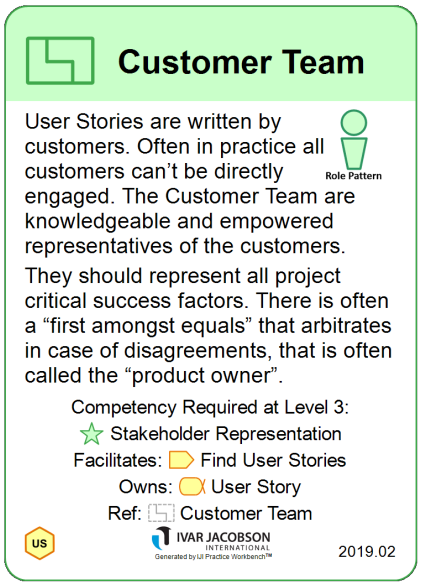
Patterns

A **Pattern** is a generic mechanism for describing practice guidance that may relate to many other associated Essence elements.

Patterns are a flexible and invaluable way to provide additional supporting guidance of any kind that relates to the other Practice elements, and how these elements work together.

Examples of common types and usages of Patterns include:

- Roles – such as this *Customer Team* example which is defined by referencing required Competencies and Competency Levels, and defining their responsibilities with respect to the Activities, Alphas and Work Products
- Milestones – which can be defined in terms of the set of Alpha States and Checklist Items that must be achieved
- Games – that use different combinations of elements of the Essence Kernel and/or Essence Practices to help teams to plan, do, assess and adapt their endeavors
- Techniques – such as *Brainstorming* or *Root-Cause Analysis* which can be described and related to the Activities that might benefit from using them.

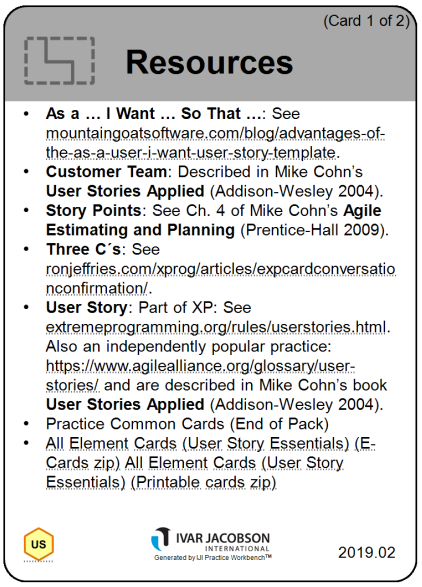


Resources

A **Resource** is a source of information or content, such as a website or a book reference.

Since an Essence Practice focuses primarily on the "bare-bone" essentials, it is useful to be able to reference other Resources where more supporting guidance can be found.

In this example for User Stories various online references, books, and other resources are listed.

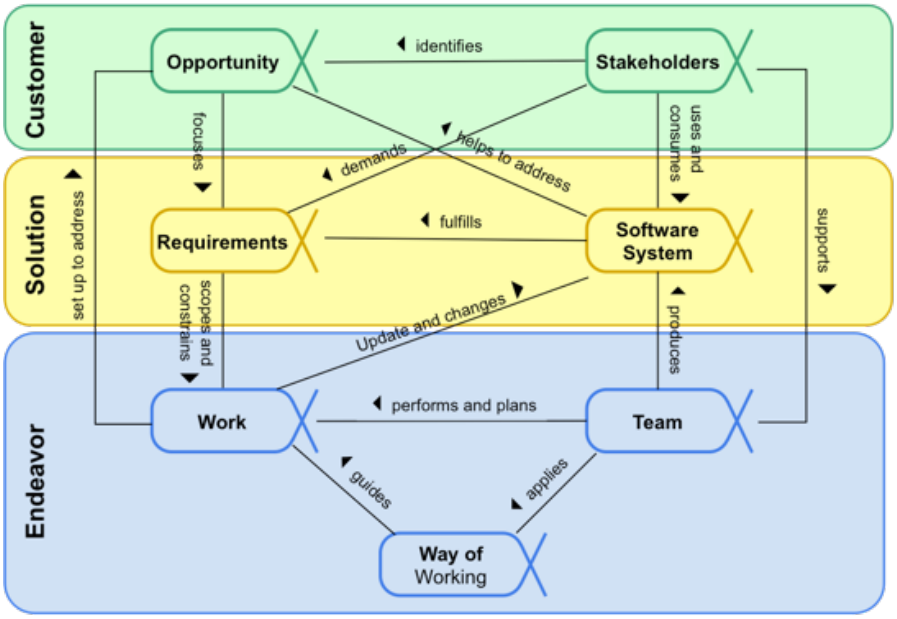


The Essence Kernel

Kernel Alphas

We have already met and defined Alphas as key aspects or elements of an endeavor that we need to progress. The Essence Kernel defines seven core, common Alphas which together:

- Capture the key concepts involved in software engineering
- Allow the progress and health of any software endeavor to be tracked and assessed
- Provide a common ground for the definition of software engineering methods and practices.



There are **customer** needs to be met

- Someone has a problem or **Opportunity** to address
- There are other **Stakeholders** who will fund, use and benefit from the solution produced

There is a **solution** to be delivered

- There are certain **Requirements** to be met
- There'll be a **Software System** to develop

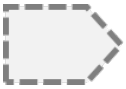
There is an **endeavor** to be undertaken

- We need to kick off the **Work** ...
- Build an empowered **Team** of good people …
- With a good, responsive **Way of Working**

As you may have already noticed, Essence subdivides the territory of software engineering into three broad Areas of Concern. Each has its own distinguishing color-coding.

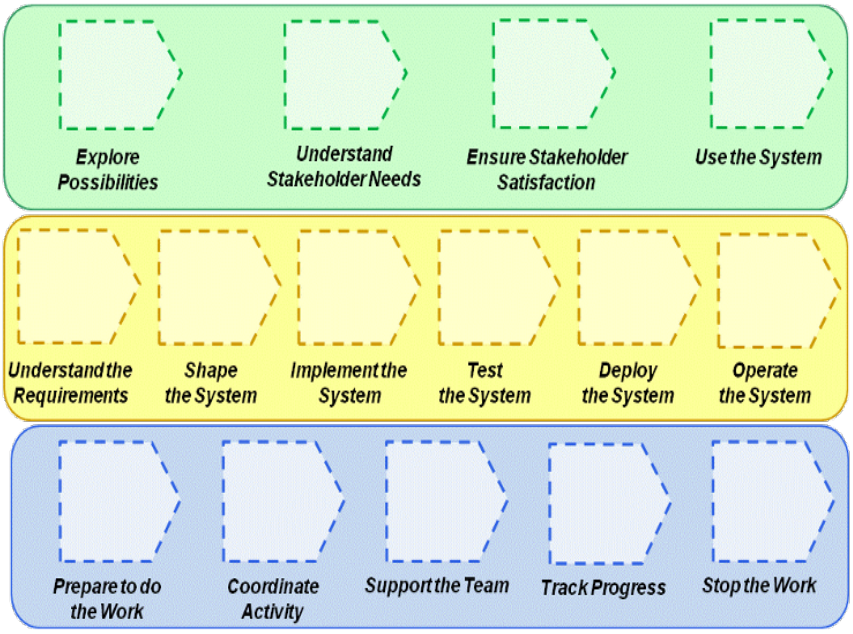
Customer	**Customer** – contains everything to do with the use and exploitation of the solution to generate value for the customers and users
Solution	**Solution** – contains everything to do the specification and development of the solution to meet the needs of the customers
Endeavor	**Endeavor** – Contains everything to do with the team, and the way that they approach their work to deliver the solution to the customer.

Activity Spaces



An **Activity Space** is a placeholder for something to be done in an endeavor such as *Understand the Requirements*.

The Essence Kernel defines a set of Activity Spaces, which together define the kinds of things that we need to do to progress any software engineering endeavor.



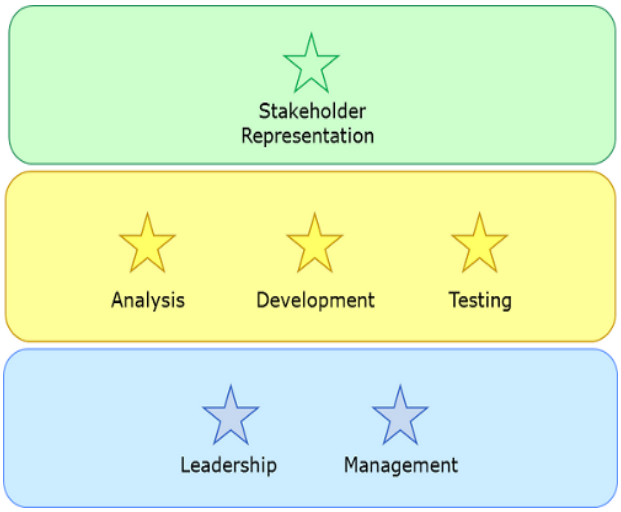
As with the Kernel Alphas, these can be used both independently and to help define practices:

- The set of Activity Spaces can be used to assess coverage and do gap analysis across a team's current way of working
- Activities in practices can be placed within the Activity Spaces to give a clear indication of what general kinds of things the practice and its activities will help the team to achieve – e.g. our earlier example *Find User Stories* Activity is in the *Understand the Requirements* Activity Space.

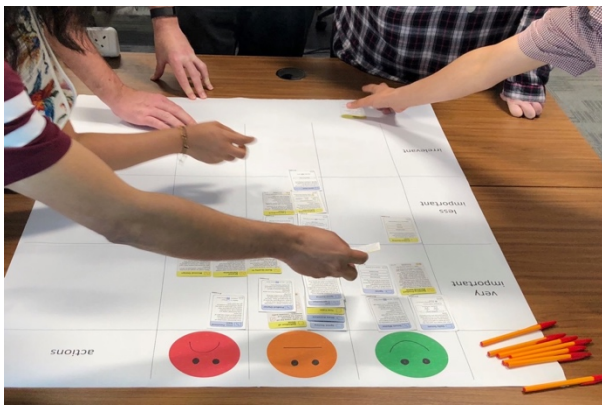
Competencies

The Essence Kernel defines six core Competencies that are the commonly needed for any software endeavor.

Practices will refer to these or can extend this set and introduce their own additional ones to address specific types of challenge. Some examples being *Operations*, *Hardware Design*, or *Coaching*.



Using Essence



Cards using the Essence Language described in this guide represent the key concepts of working practices. This opens up a whole new set of powerful games, allowing a team to reason about and improve their way of working.

The cards provide focus on the essentials to provoke the right conversations, allowing the whole team to reason and collaborate effectively.

A number of guided collaborations or 'serious games' have been evolved, refined and shared, and teams will often make up their own once they have the cards in their hands. Physically interacting with the physical cards and gameboards (or electronic equivalents) helps to establish shared understanding, builds consensus and encourages collective decision-making. Some examples of these kinds of serious games include:

- **Learning** new practices during coaching sessions and training events
- Team Retrospectives, **improving** the team's way of working (pictured above)
- Planning Activities efficiently, **coordinating** the team and other stakeholders
- **Agreeing responsibilities** within the team and with external stakeholders
- **Tracking** key progress items and status indicators using Alphas and their states
- Capturing new practices **ensuring a shared understanding** within the team.

A number of serious games can also be played with just the information provided by the Essence Kernel and are applicable regardless of the actual practices a team are using. These games are especially powerful as they give an overall holistic view of the work. This high-level view is often invisible while the teams are focused on the lower-level details. Some examples include:

- **Determining the current status** of the endeavour using the top level kernel Alphas and their States (pictured above)
- **Planning the next key achievements** at this top level
- **Comparing** the current status against a **defined high-level goal or milestone**
- **Reviewing** a way of working for **improvement opportunities and gaps**
- **Assessing the competencies** within a team against what is needed